AMENDMENTS TO THE SPECIFICATION

Please amend paragraph [0017] on page ~~4~~ 3 as follows:

a1

In applications where more ~~then~~ than one reader is in use, the host computer controlling the readers typically synchronizes the readers to prevent their collision.

Please amend paragraph [0058] on page 6 as follows:

a2

a. Reader Header Window: As mentioned, the tags periodically wake up looking for a reader. The wake up period is slightly shorter ~~then~~ than the T_{HW} time interval shown in Fig. 1. When a reader starts a new session, it transmits a header with a duration of T_{HW}. This provides the tag with the ability to detect the reader. The header transmitted by the reader contains system and reader information. The tag receives that information and typically conducts an application-specific internal process of header analysis.

Please amend paragraphs [0064] and [0065] on page 7 as follows:

a3

e. Fixed Assignment Receiving Window: When the system is steady, meaning that the tags or seals are, for the most part, not mobile, Fixed Assignment mode of operation is typically employed. In such cases the tags after switching modes respond in this window in a specific time slot. Each tag is assigned to a different specific time slot. In this window and possibly in other windows, it may be that the responses are long. If this is the case, the responses are split into packets. In a case where a tag is receiving signals from more ~~then~~ than one reader, the tag should keep track for each reader individually. In this way the tag responds to each reader in the right time slot.

f. Random Access Receiving Window: When the system is dynamic in the sense that the tags or seals repeatedly pass in and out of the reading zone of the reader, the tags respond in this window. In the random window each tag responds in a random time slot. It is also possible to have more ~~then~~ than one responding tag in the same time slot. Having more ~~then~~ than one tag in the same time slot generates a collision. It is possible to have more ~~then~~ than one transmission from a tag in this window.

a3

Retransmissions increase the probability of tag detection and are determined in the Broadcast message. For example, within the random access receiving window, communication may proceed in accordance with a slotted ALOHA procedure.

Please amend paragraph [0085] on page 10 as follows:

a4

Fig. 12 is a diagram of a preferred format for an interrogation header (IH) string transmitted by a reader as it initiates a communication session. Each reader IH string typically lasts for T_{HW} seconds and has a resolution of 1.024 msec. This time duration is synchronized with the tags. The tags wake up in a period slightly shorter ~~then~~ than T_{HW}. The default value for T_{HW} typically is approximately 3000 ms.

Please amend paragraph [0158] on page 15 as follows:

a5

Table of Fig. 19B, Row 15: Deep Sleep Wakeup Cycle (TP). To save power, in deep sleep, the wakeup cycle is longer ~~then~~ than usual. Resolution is typically 1 sec.

Please amend paragraph [0202] on page 19 as follows:

a6

In applications where tags or seals may be in reserve or standby for later usage, the tags and seals are typically deactivated to prevent their interference with the operating tags on site. In this mode the tags are in receive mode only. In this mode the wakeup cycle is 4 sec, longer ~~then~~ than the usual. "CRC" is the CRC computation for the TF & ID, whereby the tag positively identifies itself.

Please amend paragraph [0260] on page 26 as follows:

a7

The TM packet format for the Read Data Response message type is typically as illustrated in Fig. 62. In Fig. 62, the P#/P byte is the packet's serial number (P#) from total number of packets (PK). Each packet is prompted by the reader. The response instructs the tag how to proceed with the next packet. A suitable bit assignment is four high bits for P# and 4 low bits for PK. Each block of data is not more ~~then~~ than 64 bytes. A suitable maximum number of packets is 15. In case of an error with the memory data integrity, and the data is corrupted, a suitable response is sent e.g. as illustrated in Fig. 63.